实现单例模式用途广泛的有两种方式：

其中一种可以参考从boost库中摘出来的单例模式类，其保存了一个静态指针，然后通过atexit()函数注册一个销毁函数，在main函数结束后调用这个销毁函数删除静态指针。

下面介绍另一种方式：

例：程序singleton\_test

// C++中的单例模式

#include <iostream>

using std::cout;

using std::endl;

class CSingleton

{

private:

CSingleton();

CSingleton(CSingleton&); // 只声明不实现，防止在友元中拷贝

CSingleton& operator=(const CSingleton&); // 只声明不实现，防止在友元中赋值

~CSingleton();

public:

static CSingleton& GetInstance();

public:

static int count; // 用来测试是否实现了单例

};

int CSingleton::count = 0;

CSingleton::CSingleton()

{

++count;

}

CSingleton::~CSingleton()

{

}

CSingleton& CSingleton::GetInstance()

{

static CSingleton instance;

return instance; // 返回一个局部静态变量给引用

}

int main(int argc, char\* argv[])

{

CSingleton& singleton = CSingleton::GetInstance();

cout << "count = " << singleton.count << endl;

CSingleton& singleton\_1 = CSingleton::GetInstance();

cout << "count = " << singleton\_1.count << endl;

return 0;

}

输出为：

count = 1

count = 1

可见实现了单例模式，CSingleton的构造函数只调用了一次。